NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

ERROR DETECTION AND CORRECTION FOR A MULTIPLE FREQUENCY QUATERNARY PHASE SHIFT KEYED SIGNAL

by

Kevin S. Hopkins

June 1989

Co-Advisors: Daniel C. Bukofzer
Paul H. Moose

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

1a Report Security Classification: Unclassified		1b Restrictive Markings	
2a Security Classification Authority		3 Distribution Availability of Report: Approved for public release; distribution is unlimited.	
2b Declassification Downgrading Schedule			
4 Performing Organization Report Number(s)		5 Monitoring Organization Report Number(s)	
6a Name of Performing Organization: Naval Postgraduate School	6b Office Symbol (*if applicable*): 32	7a Name of Monitoring Organization: Naval Postgraduate School	
6c Address (*city, state, and ZIP code*): Monterey, CA 93943-5000		7b Address (*city, state, and ZIP code*): Monterey, CA 93943-5000	
8a Name of Funding Sponsoring Organization	8b Office Symbol (*if applicable*)	9 Procurement Instrument Identification Number	
8c Address (*city, state, and ZIP code*)		10 Source of Funding Numbers	
		Program Element No	Project No / Task No / Work Unit Accession No

11 Title (*include security classification*): ERROR DETECTION AND CORRECTION FOR A MULTIPLE FREQUENCY QUATERNARY PHASE SHIFT KEYED SIGNAL

12 Personal Author(s): Kevin S. Hopkins

13a Type of Report	13b Time Covered	14 Date of Report (*year, month, day*)	15 Page Count
Master's Thesis	From To	June 1989	60

16 Supplementary Notation: The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes			18 Subject Terms (*continue on reverse if necessary and identify by block number*)
Field	Group	Subgroup	Error Detection Correction, Coding, MFQPSK, EDC

19 Abstract (*continue on reverse if necessary and identify by block number*)

A multiple frequency quaternary phased shift keyed (MFQPSK) signaling system has been developed and experimentally tested in a controlled environment. In order to insure that the quality of the received signal is such that information recovery is possible, error detection correction (EDC) must be used. This thesis reviews various EDC coding schemes available and analyzes their application to the MFQPSK signal system. Hamming, Golay, Bose-Chaudhuri-Hocquenghem (BCH), Reed-Solomon (R-S) block codes as well as convolutional codes are presented and analyzed in the context of specific MFQPSK system parameters. A computer program was developed in order to compute bit error probabilities as a function of signal-to-noise ratio. Results demonstrate that various EDC schemes are suitable for the MFQPSK signal structure, and that significant performance improvements are possible with the use of certain error correction codes.

20 Distribution Availability of Abstract: ☒ unclassified unlimited ☐ same as report ☐ DTIC users	21 Abstract Security Classification: Unclassified	
22a Name of Responsible Individual: Daniel C. Bukofzer	22b Telephone (*include Area code*): (408) 646-2859	22c Office Symbol: 62Bh

DD FORM 1473,84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

Approved for public release; distribution is unlimited.

ERROR DETECTION AND CORRECTION FOR A MULTIPLE FREQUENCY QUATERNARY PHASE SHIFT KEYED SIGNAL

by

Kevin S. Hopkins
Lieutenant, United States Navy
B.S., United States Naval Academy, 1981

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 1989

ABSTRACT

A multiple frequency quaternary phased shift keyed (MFQPSK) signaling system has been developed and experimentally tested in a controlled environment. In order to insure that the quality of the received signal is such that information recovery is possible, error detection/correction (EDC) must be used. This thesis reviews various EDC coding schemes available and analyzes their application to the MFQPSK signal system. Hamming, Golay, Bose-Chaudhuri-Hocquenghem (BCH), Reed-Solomon (R-S) block codes as well as convolutional codes are presented and analyzed in the context of specific MFQPSK system parameters. A computer program was developed in order to compute bit error probabilities as a function of signal-to-noise ratio. Results demonstrate that various EDC schemes are suitable for the MFQPSK signal structure, and that significant performance improvements are possible with the use of certain error correction codes.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

A. CONCEPT OF OPERATIONS

Modulation is the process by which signals are transformed into waveforms tailored to the characteristics of the transmission channel. The medium for the data transmission may be free space, a wire, an optical fiber or underwater acoustics, each possibly requiring a different modulation technique. It is not desireable to have to design communications links around the type of modulation dictated by the propogation medium. A single modulation technique that is flexible, can be used for analog and digital signals, is not channel restrictive, and can emulate existing modulation techniques would be ideal.

Multi-Frequency Modulation (MFM) [Ref. 1] is a technique that embodies some of the above characteristics. Developed at the Naval Postgraduate School, MFM was designed for computer-to-computer communication links and information exchange networks. Applications of MFM to Navy satellite communication, ship-to-ship/shore communication and acoustic communication are being considered. MFM systems operating at audio frequencies are particularly suited to acoustic communications. Concurrent research with Naval Ocean Systems Center (NOSC), San Diego is ongoing in this area. More specifically, a Multi-Frequency Quaternary Phased Shift Keyed (MFQPSK) signaling system is being built and tested by NOSC with Naval Postgraduate School assistance.

B. PREVIOUS RESEARCH

The MFQPSK signaling system is well suited for high data rate acoustic communications [Ref. 2]. Research in MFQPSK signaling systems at the Naval Postgraduate School is currently in progress. However, past efforts in this area of research have primarily focused on the generation of the signal itself through Fast Fourier Transform techniques. Limited attention has been given to the quality of the received signal. That is, error detection/correction (EDC) has played a minor role in signal development to date. During an initial feasibility study, a single error correction/double error detection radix-5 Hamming code was proposed [Ref. 3]. However, since this initial proposal, EDC coding has been generally neglected, although it is seen as a necessary tool for the successful use of MFQPSK.

C. OVERVIEW

The purpose of this study is to explore various coding schemes and evaluate their effectiveness when used in conjunction with MFQPSK signaling. Three coding schemes in particular (BCH, Reed-Solomon and convolutional) are singled out and explored in depth. An algorithm was developed to evaluate the performance of these codes and their application to this project. Chapter Two describes the MFQPSK signal, so as to provide a brief conceptual understanding of the signal without going into the in-depth explanation that can be found in References 1 and 4. In Chapter Three, derivation of the basic probability equations and explanation of some of the channel variables that affect signal transmission are presented. Chapter Four presents a detailed discussion of various coding schemes and their associated bit error performance. Some schemes (such as Hamming and Golay codes) are presented for completeness in describing the fundamentals of coding theory but are not analyzed in depth in so far as their application to MFQPSK signaling. Chapter Five is devoted to the application of error performance characteristics of various coding schemes to the MFQPSK signaling system. Chapter 6 concludes this study by singling out a particular coding scheme that is well-suited for the MFQPSK signaling and suggesting areas of further research.

II. THE MFQPSK SIGNAL

The specific design parameters of the MFQPSK signal used in this research project are not necessarily design constraints on the EDC coding scheme. However, they are included here in order to provide the reader with a better understanding of the MFQPSK signal and the overall MFQPSK communication system operation. By modulating frequencies with phase information and using frequency to time domain transformations via the Fast Fourier Transform (FFT), the MFQPSK transmission signal can be generated.

Figure 1 shows the basic structure of the MFQPSK signal.

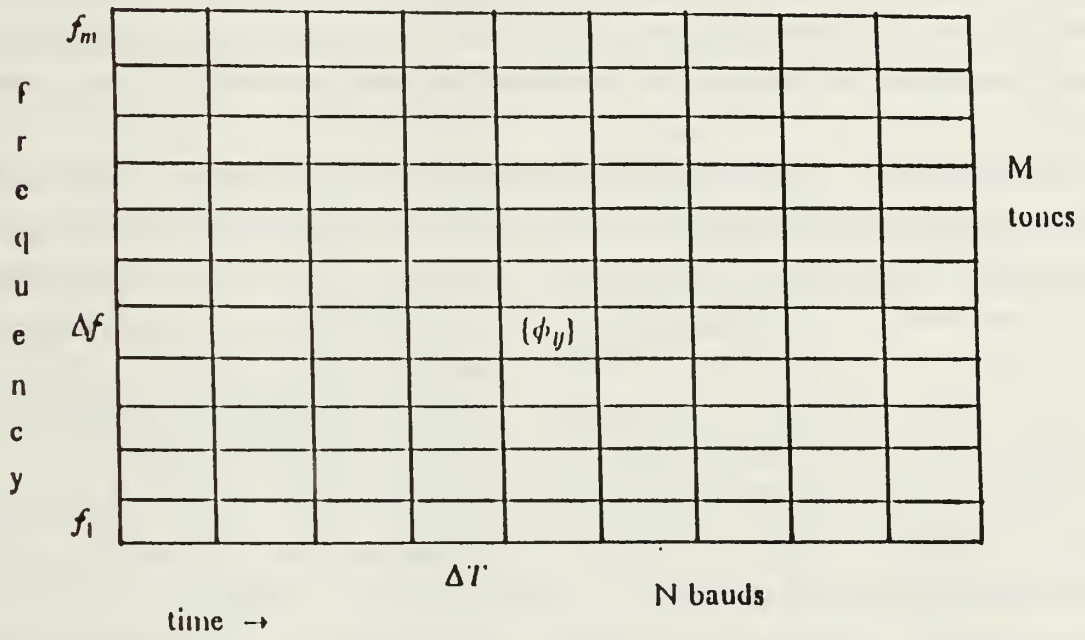

Figure 1. The Structure of the MFQPSK Signal

Frequency is plotted on the vertical axis depicting M tones, Δf Hz apart. Time is plotted on the horizontal axis showing N bauds each of length ΔT seconds. The symbols,

$\{\phi_{ij}\}$ represent the phase of the j^{th} tone in the i^{th} baud. In quadrature phase (2 bits/symbol), $\{\phi_{ij}\} = \left\{ \pm \frac{\pi}{4}, \pm \frac{3\pi}{4} \right\}$.

In the MFQPSK signaling system, the $\{\phi_{ij}\}$ components represent unique time/frequency phases. These components and their complex conjugate at the image frequencies are loaded into a two dimensional array that is used as an input to an Inverse Fast Fourier Transform (IFFT). The image frequency components are needed to insure that the corresponding time domain signal is entirely real. Thus, the IFFT generates a collection of real, time domain signal sequences that are time samples of the analog transmit signal. Processing these through a D/A converter completes the generation of the i^{th} baud. The total signal packet is completed by an N-fold repetition of this frequency-to-time domain transformation.

A complete data packet, P, is made up of M tones in N bauds. (It is not necessary that all of the frequency bins contain data information. The j^{th} tone could be used for synchronization for example.) Since one of four phases is determined by a di-bit (a sequence of two binary bits) there will be $2NM$ bits in each packet. The tone spacing is the inverse of the baud length. By increasing baud lengths, more tones are used and vice-versa.

Demodulation of the MFQPSK signal is accomplished by a process that is just the opposite of the method of generation. The time domain signal is filtered, sampled and converted to a digital format through an A/D converter. This sequence is partitioned into N sequences corresponding to the N bauds and loaded into a complex valued array with imaginary parts set to zero. This array becomes the input to an FFT which generates a complex valued array containing the phases of the original transmitted baud. Computing this N-fold times, will generate the total transmitted data packet.

This brief explanation is provided for conceptual purposes only and intentionally does not address the problems of synchronization, channel noise, multipath propagation and non-linear characteristics of the supporting equipment. Individually, these are subjects of various research projects currently associated with the MFQPSK signaling system.

III. CHANNEL/CODING CONSIDERATIONS

A. CHANNEL CONSIDERATIONS

The acoustic properties of the ocean constrain the MFQPSK signal in this application in various ways. For instance, since short signal wavelengths are absorbed more readily than longer wavelengths, the signal transmission bandwidth is limited by the desired maximum range of communications. This will, in turn, constrain the maximum rate at which data can be transmitted. Another channel consideration concerns the multipath characteristics of acoustic propagation. Surface and sea bottom reflections as well as sound channels may cause the signal to propagate through various paths with different delays. Doppler shift due to the movement of the source and the random dynamics of the sound velocity pose additional problems.

Acoustic channel modeling for the MFQPSK signaling system is a separate project being explored at the Naval Postgraduate School. Its purpose is to identify and model important the variables of acoustic signal transmission. Also included in the model are the symbol energy-to-noise spectral density ratio E_s/N_o, the bandwidth and data rate. (Since E_s/N_o is actually a signal-to-noise ratio it will be referred to simply as SNR throughout this work.) Once complete, coding gains can be computed based on this model as a function of SNR, baud length, doppler mismatch, etc.

B. CODING CONSIDERATIONS

In this signaling system, the purpose is to transmit digital information at the maximum possible rate with a fixed signal power while maintaining adequate error performance. Varying definitions of adequate error performance has led to the specification of many EDC coding schemes, each exploiting some feature of the transmitter-channel-receiver trilogy. If the medium introduces high-level constant noise, increasing transmitted power may be the only means to compensate for the noise and ensure adequate receiver performance. Perhaps full replication of the transmitted signal is not needed on the receiving end. In this case, a lower transmitter power may be used and the transmitted signal extrapolated from what is received. Additional signal degradation from such variables as mulitpath propagation, fading, and doppler shift further reduce the received SNR and/or create decoding errors and amplify the need for error detection and correction.

The MFQPSK signal is to be encoded from a binary data stream. Figure 2 shows the block structure of a binary data stream, how it is partitioned into information and parity bits and illustrates the meaning of a di-bit.

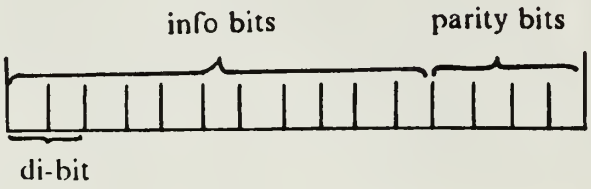

Figure 2. Block Coding of a Binary Data Stream

Due to the randomness of phase assignments, it may be assumed that any incorrect decisions of the receiver will have equal probabilities.

In light of the channel considerations previously discussed, a coding scheme must be identified which will provide adequate error performance. However, since all these studies are not yet complete, all performance graphs in this thesis will be plotted showing probability of error as a function of SNR.

1. Probability Development

In this study, two error probabilities will be continually compared. The first will be that of a MFQPSK signal which is not EDC coded. In this case, bit error probabilities will be dependent on the type of demodulation used. Secondly, EDC coded received signal error performance for particular coding schemes will be developed and compared to each other in order to evaluate which coding schemes may be most suitable for an MFQPSK signaling system.

a. Uncoded transmission

In determining the probability of error for an uncoded bit stream, the primary consideration is the modulation type. Since there is no coding involved, this will specify the performance of the signaling system. Figure 3 illustrates a QPSK signaling constellation developed from Reference 5 .

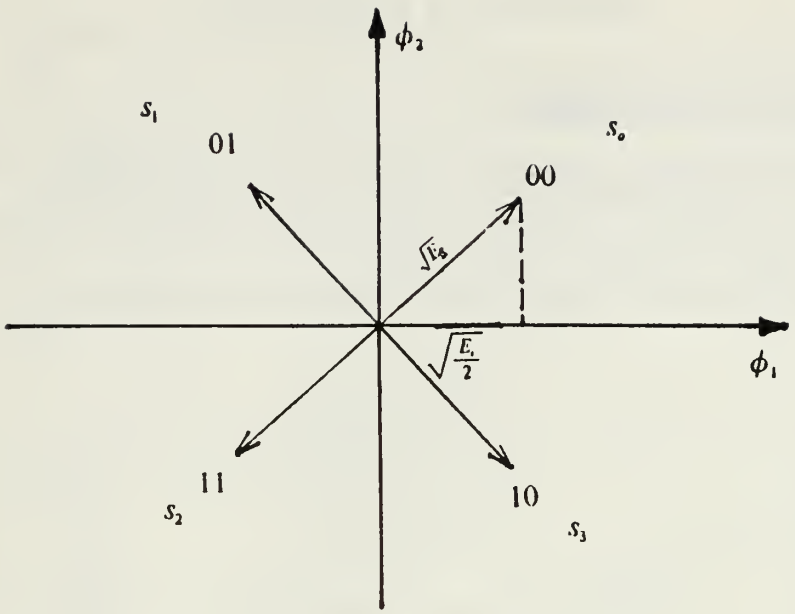

Figure 3. QPSK Signal Constellation Diagram

Receiving two bits of information, left bit and right bit (L_b and R_b, respectively), the probability that the left bit was received correctly (L_bC) in additive white Gaussian noise given that signal s_o was transmitted is:

$$Pr\{L_bC/s_o\} = Pr\{L_bC/s_1\} = 1 - Q\left(\sqrt{\frac{E_s}{N_o}}\right) \tag{3.1}$$

$$= Pr\{L_bC\} \tag{3.1a}$$

The function Q(x) is the complementary error function or co-error function and is defined as:

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty \exp\left(-\frac{u^2}{2}\right) du \tag{3.2}$$

Equation (3.1a) now represents the unconditional probability that L_b is correct. This is irrespective of what signal was transmitted. Similarly, the unconditional probability that R_b is correct is identical. Thus,

$$Pr\{bit\ correct\} = 1 - Q\left(\sqrt{\frac{E_s}{N_o}}\right) \tag{3.3}$$

Therefore, the bit error probability is

$$p = Pr\{bit\ incorrect\} = 1 - Pr\{bit\ correct\} = Q\left(\sqrt{\frac{E_s}{N_o}}\right) \tag{3.4}$$

A graph of p as a function of E_s/N_o, denoted SNR, is shown in Figure 4.

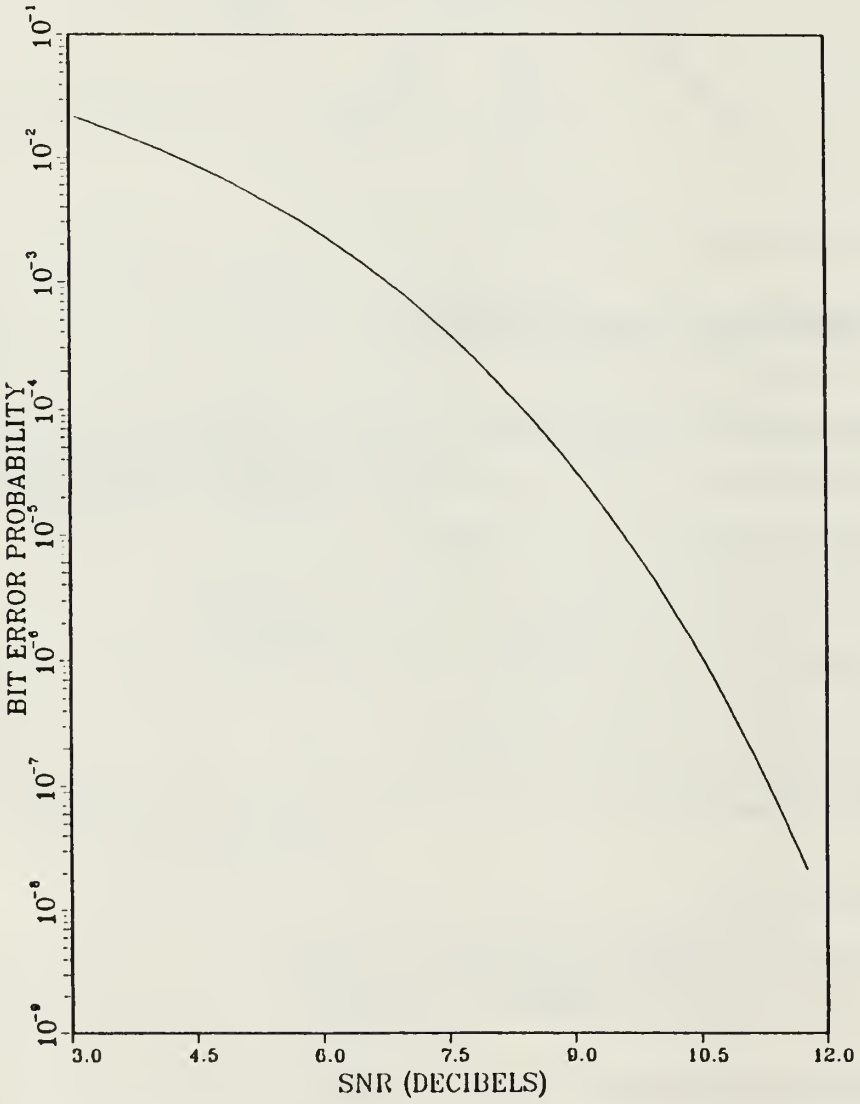

Figure 4. Bit Error Probability for Uncoded QPSK Signals

b. Received EDC coded transmission

When a bit stream is transmitted over a noisy channel, errors will occur causing the transmitted and received bit streams to differ. Given that a specific bit was transmitted, the probability that it is received in error is p, and the probability that it is correctly received is $(1 - p)$. Given a message consisting of a string of n bits, the probability of no error in all n bits is the individual correct probabilities raised to the number of n bits, namely,

$$(1-p)^n \tag{3.5}$$

and the probability of a single error in the n bits is:

$$np(1-p)^{n-1} \tag{3.6}$$

Similarly, the probability of exactly two errors in the n bits is:

$$\frac{n(n-1)}{2!}p^2(1-p)^{n-2} \tag{3.7}$$

Equations (3.6) and (3.7) represent exact values for probability of error. This is for the case that exactly n or fewer errors are detected and corrected. Codes that do this are called perfect codes. As will be explained in Chapter 4, there are very few perfect codes: the Hamming ($t = 1$), the Golay ($t = 3$) and a ternary Golay code are examples. Other t-error correcting codes can correct all occurrences of t errors and varying percentages of more than t errors. When computing P_B for non-perfect codes, it is understood that the code is capable of correcting all occurrances of $j \leq t$ errors. However, more that t errors can occur, that is $j = t + 1$, resulting in decoded errors. Therefore, by summing the total probability of j errors in the n positions and dividing by the number of bits leads an upper bound on P_B, the probability of bit error of t-error correcting codes:

$$P_B \leq \frac{1}{n}\sum_{j=t+1}^{n} j \binom{n}{j} p^j (1-p)^{n-j} \tag{3.8}$$

where $p = Q\left(\sqrt{SNR}\right)$. A plot of equation (3.9) appears as Figure 5.

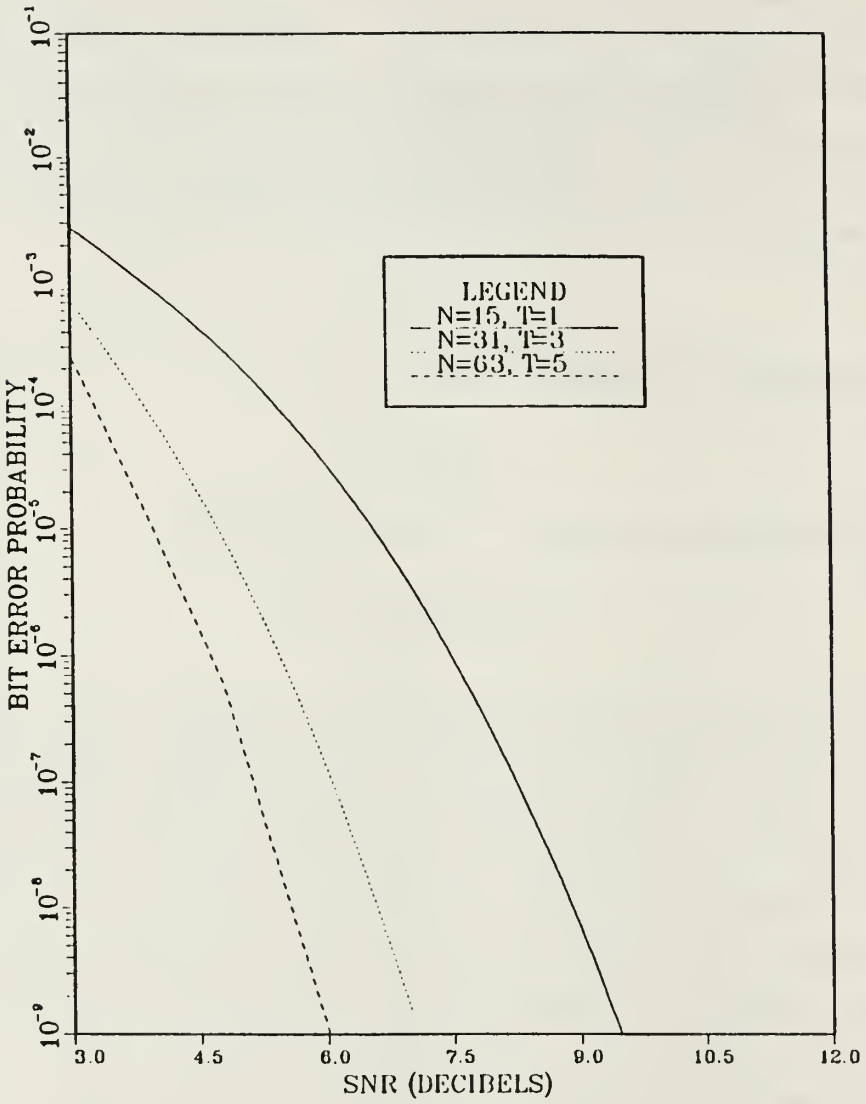

Figure 5. General Bit Error Probability for Coded Signals

IV. ERROR DETECTION/CORRECTION CODES

A. FUNDAMENTAL CONCEPTS

Error detection/correction (EDC) coding provides error control in digital information transmission. An encoder adds redundancy to a transmitted message in such a manner that, upon decoding, the correct digital information, even in the presence of channel-induced errors, can be determined. Redundancy is added in such a way that the decoder decisions can be based on several received bits rather than on one, as in uncoded systems.

Error control can be accomplished using forward error correction (FEC), automatic repeat request (ARQ), or a variety of hybrid FEC-ARQ approaches. Using FEC, the decoder corrects as many channel errors as possible. This is done by generating an estimate of the transmitted sequence within the limitations of the code. Using ARQ, the decoder detects errors and prompts the transmitter to retransmit as required. Because of the duplex nature of the ARQ procedure, it was not considered viable for this project. Therfore FEC coding only is being considered here.

FEC codes can be classified in two general categories, block and convolutional.

1. Block Codes

Block codes process data in blocks that are independent from each other. The encoder transforms a block of k message bits into a block of n binary coded bits, where $n > k$. The ratio of k/n is called the code rate R, where $0 < R < 1$. Block codes are expressed in terms of n and k such as a (n, k) code. A block code represents a one-to-one transformation where 2^k information k-tuples are uniquely mapped into a set of 2^k codeword n-tuples.

A wide variety of linear block codes exist. The ones that will be considered here as they may apply to this project include: the Hamming, Golay, Bose-Chaudhuri-Hocquenghem (BCH), and Reed-Solomon (R-S) codes.

2. Convolutional Codes.

Convolutional codes provide n output coded bits for each group of k input bits. However any n-coded bit convolutional encoder output depends not only on the last set of k input bits but also on several preceeding sets of input bits. Convolutional encoders are described by n, k and K. The code rate, R, is defined by k/n , and K, known as the constraint length, is the number of stages in the shift register used. Shift registers are

used to process the bit stream. As a bit is fed into the shift register, that particular block of bits in the shift register are modulo-2 added in a predetermined manner, described by so-called generating polynomials, and the encoded bit stream output is taken from the modulo-2 adders. The next bit is then fed into the shift register causing all previous bits to shift one stage and the output is again taken from the modulo-2 adders. This process is repeated until all the information bits are processed through the shift register. Convolutional codes are expressed either as (n, k, K) or (R, K). As an example, a (1/2, 3) convolutional encoder is shown in Figure 6.

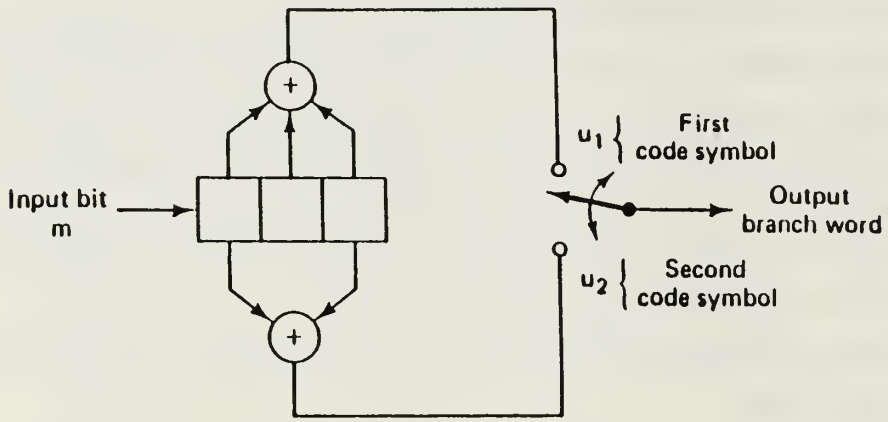

Figure 6. Convolutional Encoder: (Rate = 1/2, K = 3, reproduced from page 319 of Reference [6])

B. HAMMING CODES

Hamming codes are a class of block codes used for error detection and/or correction applications. They are also members of a group of codes called perfect codes. A perfect code is one in which all the vectors described by the code are contained within a sphere of radius $t = [(d_{min} - 1)/2]$ about the codewords. Here t represents the number of errors the code can correct and d_{min} is the minimum distance between codewords of the code. Minimum distance is an integral part of determining the error correcting capability of block codes. To define d_{min}, consider the following: if u and v are two vectors of a code

then, by definition, $\mathbf{u} \oplus \mathbf{v}$ must also be a code vector where code vectors represent a bit sequence of length n and $\oplus$ indicates modulo-2 addition. The Hamming distance between any two code vectors is equal to the number of places where the two code vectors disagree. This corresponds to the weight (i.e., number of 1's) of $\mathbf{u} \oplus \mathbf{v}$. Thuis d_{min} is the minimum weight of all non-zero code vectors. A Hamming code has ($n = 2^m$-1, $k = 2^m$ -1-m, $d_{min} = 3$). Therefore, one can see the (7, 4) Hamming code has a minimum weight of $d_{min} = 3$ and therefore, has a single error correcting capability. The binary Hamming (7, 4) code is also perfect since 2^3 (the number of vectors in a sphere of radius 1 about a codeword) times 2^4 (number of codewords) equals 2^7, the total number of vectors [Ref. 7].

Hamming codes are developed around some basic principles:

- All noise is modeled as channel noise.
- The channel is a binary symmetric channel.
- In all computations, modulo base must equal radix base (in most cases this will be modulo 2).
- Hard decision decoding is used.[1]

Parity check matrix and syndrome development are necessary when evaluating block codes. The parity check matrix **H** consists of all nonzero m-tuples as its columns, arranged in the following form:

$$\mathbf{H} = [\mathbf{I}\ \mathbf{Q}] \tag{4.1}$$

where **I** is an $m \times m$ identity matrix and the submatrix **Q** consists of 2^m-m-1 columns which are m-tuples of weight 2 or more. As an example, with $m = 3$,

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix} \tag{4.2}$$

The coded vector **c** of a Hamming code is constructed from binary representations of position locations. All odd numbered positions end with a 1 when represented in binary form. Therefore, a parity check is used to cover the odd positions 1, 3, 5, 7, ... Similarly, the second parity check covers positions 2, 3, 6, 7, 10, 11 all these positions

---

1 Hard decisions refer to the cases in which no confidence associated with the bit decision is known.

have a 1 in the second-lowest position in its binary form. The third parity check will cover positions 4, 5, 6, 7, 12, 13, 14, 15, ...

If the information to be transmitted contains four bits, say [1 0 1 0], the coded vector will be constructed as [- - 1 - 0 1 0] where the (-) are used as parity checks. The total number of odd positions 1's are modulo-2 added and the result entered in the first position. Now c = [1 - 1 - 0 1 0]. Next, the second parity check is entered in position two as the modulo-2 sum of the digits in positions 2, 3, 6, and 7. The coded vector is now [1 0 1 - 0 1 0]. Finally, by modulo-2 adding the 1's in positions 4, 5, 6, and 7 the third parity check is accomplished and **c** is completed as [1 0 1 1 0 1 0].

Given a coded transmitted vector **c**, where **c** is the coded bit stream of information bits and error correction coding, and a received vector **r**, Reference [8] defines the syndrome as $\mathbf{S} = \mathbf{rH}$. The syndrome will be used to locate error positions. If no transmission errors occur then, since $\mathbf{Hc}^T = 0, \mathbf{Hr}^T = 0$ because $\mathbf{r} = \mathbf{c}$.

To continue the above example, let $\mathbf{c} = \mathbf{r} =$ [1 0 1 1 0 1 0]. This is the desired coded transmission and contains no errors. Therefore, the syndrome

$$\mathbf{S} = \mathbf{Hr}^T = 0$$

$$\mathbf{S} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = 0 \qquad (4.3)$$

If a single error were introduced, the received vector would be: $\mathbf{r} = (\mathbf{c} + \mathbf{e})$ where **c** represents the transmitted vector and **e** represents an introduced error. Computing the **S** vector, $\mathbf{S} = \mathbf{Hc}^T + \mathbf{He}^T$ and knowing that $\mathbf{Hc}^T = 0$, $\mathbf{S} = \mathbf{He}^T$. The syndrome depends on the error vector and not on the codeword sent. The syndrome is a vector which represents the column where the errors occur.

Now using the previous example, let the transmitted vector be

$$\mathbf{c} = [1011010] \qquad (4.4)$$

and let **e** = [0 0 0 1 0 0 0], so that **r** = [1 0 1 0 0 1 0].

Since

$$\mathbf{H}\,\mathbf{c}^{\mathrm{T}} + \mathbf{H}\,\mathbf{e}^{\mathrm{T}} = \mathbf{S}$$

and

$$\mathbf{H}\mathbf{c}^{T} = \mathbf{0},$$

then,

$$\mathbf{S} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \tag{4.5}$$

Comparing the syndrome to the **H** matrix shows that entries in the fourth column of **H** are identical to the syndrome thus indicating an error in the fourth position of the received vector. By inverting the digit in this position, a correction to the received vector is accomplished.

Hamming codes can easily be extended to double error detection single error correction codes by adding a parity check as the top row of the matrix. For example, the **H** matrix of Equation (4.2) results in

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} \tag{4.6}$$

The usefulness of this extension is limited as the extended Hamming code can only detect the existence of a double error, and still only correct single errors.

Hamming codes were developed for self-checking of the Model 5 Relay Computers built by Bell Telephone Laboratories in the late 1940's [Ref. 8]. Because of computer infancy at the time of the code's creation, Hamming codes are relatively simple and of

limited capabilities by today's standards. However, they serve as the cornerstone of EDC coding and are described in this report for completeness.

The initial feasibility study of the MFQPSK signal included a radix-5 Hamming code for each of the frequency bins [Ref. 3]. Using modulo-5 algebra and a two-dimensional matrix, the radix-5 code proved useful. However, as with all Hamming codes, it was only capable of single error correction.

C. GOLAY CODES

The next extension of the Hamming code is the Golay code. Like the Hamming code, the (23, 12) Golay code is also a perfect code. In fact, the Hamming and Golay codes are the only known binary perfect codes.[2] Because the code rate ($k/n = 12/23$), is awkard to work with, a parity bit is added creating an extended (24, 12) Golay code. This added parity bit increases d_{min} from 7 to 8. Both Golay codes possess triple error correcting capabilities. The extended Golay code is more powerful than the Hamming codes but also carries drawbacks: a more complicated decoder, a lower code rate and, with it, greater bandwidth expansion.

Although Golay codes' error performances are superior to Hamming codes, from a practical point of view, Golay codes are limited by their short block lengths and have limited practical application when considering the availability of other more powerful codes. The Golay code's uniqueness as a perfect code (or *quasi-perfect* for the extended Golay code) carries with it many mathematical properties which are used in coding theory to group theory and other, more esoteric, mathematical topics.

D. BCH CODES

Of the many classes and subclasses of random error-correcting codes proposed to date, the class discovered by Hocquenghem in 1959 and independently by Bose and Chaudhuri in 1960 is the most extensive and powerful one. The Bose-Chaudhuri-Hocquenghem (BCH) codes are a generalization of the Hamming codes but allow multiple-error correction, while providing a large selection of block lengths, code rates, and error correcting capability. Because of this flexibility, there are many BCH code implementation algorithms. At block lengths of a few hundred, the BCH codes outperform many other block codes with similar block length and code rate [Ref. 6].

For any positive integer m and t ($t < 2^{m-1}$), there exists a BCH code with the following parameters:

---

2 There is a perfect ternary (11, 6) Golay code but it will not be discussed in this research.

Block length:	$n = 2^m - 1$
Number of parity-check digits:	$n - k \leq mt$
Minimum distance:	$d_{min} \geq 2t + 1$

This code is capable of correcting any combination of t or fewer errors in a block of $n = 2^m - 1$ digits.

The alphabet of a BCH code for $n = (2^m - 1)$ may be represented by the elements of the appropriate Galois Field, GF(2^m), whose primitive element is α. The generator polynomial, $g(\cdot)$ of the t-correcting BCH code is given by the least common multiple (LCM) of $M_1(X), M_2(X), ..., M_{2t}(X)$. That is,

$$g(X) = LCM[M_1(X), M_2(X), ..., M_{2t}(X)] \tag{4.7}$$

where $M_i(X)$ is the minimum polynomial of of α^i, i = 1, 2, ..., $2t$, each unique to $GF(2^m)$.[3] Since the minimum polynomials for $\alpha^2, \alpha^4, ..., \alpha^{2t}$ (all even powers of α) are the same as those of $\alpha, \alpha^2, \alpha^3, \ldots$, the generator polynomial is reduced to:

$$g(X) = LCM[M_1(X), M_3(X), ..., M_{2t-1}(X)] \tag{4.8}$$

Values for $M_i(X)$ are tabulated in texts and technical reports based on values of m for specific GF(2^m).

Consider a (15, 7) double-error correcting BCH binary code for which $m = 4$. The generating polynomial is:

$$g(X) = LCM[M_1(X), M_3(X)] \tag{4.9}$$

$$= M_1(X)M_3(X)$$

$$= (1 + X + X^4)(1 + X + X^2 + X^3 + X^4)$$

$$= 1 + X^4 + X^6 + X^7 + X^8 \tag{4.9a}$$

Similarly, for a (15, 5) triple-error correcting BCH binary code over GF(2^4), the generating polynomial is:

$$g(X) = LCM[M_1(X), M_3(X), M_5(X)] \tag{4.10}$$

---

3 The minimum polynomial over $GF(2^m)$ of β is the lowest degree monic polynomial $M(\beta)$ with coefficients from $GF(2^m)$ such that $M(\beta) = 0$.

$$= M_1(X)M_3(X)M_5(X)$$

$$= (1 + X + X^4)(1 + X + X^2 + X^3 + X^4)(1 + X + X^2)$$

$$= 1 + X + X^2 + X^4 + X^4 + X^5 + X^8 + X^{10} \qquad (4.10a)$$

1. Use of Generator Polynomials

Generator polynomials are used in the formulation of a generator matrix, **G**, where $\mathbf{G} = [\mathbf{I}_k \ \mathbf{P}]$ and a parity check matrix **H**, where $\mathbf{H} = [\mathbf{I}_{n-k} \ \mathbf{P}^T]$. A code vector **c**, is obtained by multiplying the message vector **m** and the generator matrix **G**, namely

$$\mathbf{c} = \mathbf{mG} \qquad (4.11)$$

The received vector **r** will consist of:

$$\mathbf{r} = \mathbf{c} + \mathbf{e} \qquad (4.12)$$

(If there are no errors, $\mathbf{e} = 0$.) Obtain the syndrome **S**,

$$\mathbf{S} = \mathbf{rH}^T = \mathbf{eH}^T \qquad (4.13)$$

Then by inverting the digits at the locations specified by the syndrome, the decoded vector **d** is obtained, where **d** corresponds to **c** so long as there are t or fewer errors.

2. Decoding BCH Codes

Chapter 6 of Reference [9] presents a detailed discussion of decoding BCH codes. The highlights of that discussion are presented here. Consider a received vector $\mathbf{r} = \mathbf{c}(X) + \mathbf{e}(X)$ as before. Since $\mathbf{g}(X)$ is a factor in $\mathbf{c}(X)$ and $\alpha, \alpha^2, \ldots, \alpha^{2t}$ are the roots of $\mathbf{g}(X)$ then,

$$\mathbf{r}(\alpha^i) = \mathbf{c}(\alpha^i) + \mathbf{e}(\alpha^i) \qquad (4.14)$$

$$= \mathbf{e}(\alpha^i) \quad i = 1, 2, \ldots 2t \qquad (4.14a)$$

since $\mathbf{c}(\alpha^i) = 0$. Define now the syndrome vector

$$\mathbf{S}_i = \mathbf{r}(\alpha^i) \qquad (4.15)$$

$$= \mathbf{r}_0 + \mathbf{r}_1\alpha^i + \mathbf{r}_2(\alpha^i)^2 + \ldots + \mathbf{r}_{n-1}(\alpha^i)^{n-1} \quad i = 1, 2, \ldots, 2t \qquad (4.15a)$$

where $r_0, r_1, \ldots, r_{n-1}$ are the coefficients of $\mathbf{r}(X)$. Thus,

$$\mathbf{S}_i = \mathbf{e}(\alpha^i) \tag{4.16}$$

$$= \mathbf{e}_0 + \mathbf{e}_1\alpha^2 + \mathbf{e}_2(\alpha^i)^2 + \ldots + \mathbf{e}^{n-1}(\alpha^i)^{n-1} \tag{4.16a}$$

In order to simplyfy the notation, assume p errors have occurred where $0 < p \leq t$ at locations given by $X_1, X_2, \ldots, X_p$ and magnitudes (binary digits) given by $Y_1, Y_2, \ldots, Y_p$. Then the syndrome vectors may be written:

$$\begin{aligned}
\mathbf{S}_1 &= Y_1X_1 + Y_2X_2 + \ldots + Y_pX_p \\
\mathbf{S}_2 &= Y_1X_1^2 + Y_2X_2^2 + \ldots + Y_pX_p^2 \\
&\cdot \quad \cdot \quad \cdot \\
&\cdot \quad \cdot \quad \cdot \\
&\cdot \quad \cdot \quad \cdot \\
\mathbf{S}_{2t} &= Y_1X_1^2 + Y_2X_2^{2t} + \ldots + Y_pX_p^{2t}
\end{aligned} \tag{4.17}$$

The difficulty in solving this set of equations is due to their non-linearity and non-unique solution. By defining intermediate variables and an error locator polynomial, $\lambda(z)$ given by

$$\lambda(z) = \lambda_p z^p + \lambda_{p-1}z^{p-1} + \ldots + \lambda_1 z + 1 \tag{4.18}$$

which has zeros at the inverse error locations X_l^{-1}, l= 1, 2, ..., p. Then,

$$\lambda(z) = (1 - zX_1)(1 - zX_2)\ldots(1 - zX_p) \tag{4.19}$$

The zeros of this equation lead to the error locations and are equal to the coefficients, $\lambda_p, \lambda_{p-1}, \ldots, \lambda_1$ of $\lambda(z)$. The coefficients and syndromes are related by the matrix equation:

$$\mathbf{M} = \begin{bmatrix} S_1 & S_2 & S_3 & \cdot & S_{p+1} \\ S_2 & S_3 & S_4 & & S_{p+2} \\ S_3 & S_4 & S_5 & \cdot & S_{p+3} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ S_p & S_{p+1} & S_{p+2} & \cdot & S_{2p-1} \end{bmatrix} \begin{bmatrix} \lambda_p \\ \lambda_{p-1} \\ \lambda_{p-2} \\ \cdot \\ \cdot \\ \cdot \\ \lambda_1 \end{bmatrix} = \begin{bmatrix} -S_{p+1} \\ -S_{p+2} \\ -S_{p+3} \\ \cdot \\ \cdot \\ \cdot \\ -S_{2p} \end{bmatrix} \tag{4.20}$$

Provided that this matrix is nonsingular, and therefore invertible, the above equation can be solved. Proof of nonsingularity is found on page 169 in Reference 10.

Consider for example a (15, 7) double error correcting BCH code in GF(2^4) and modulo polynomial given by $X^4 + X + 1$, so that

$$\mathbf{g}(X) = X^8 + X^7 + X^6 + X^4 + 1 \tag{4.21}$$

If the message, m(X) = (1 + X), then the transmitted vector is

$$\mathbf{c}(X) = X^9 + X^6 + X^5 + X^4 + 1. \tag{4.22}$$

The received vector is now assumed to be

$$\mathbf{r}(X) = X^9 + X^8 + X^6 + X^5 + X^4 + X^3 + 1 \tag{4.23}$$

so that

$$\mathbf{e}(X) = X^8 + X^3 \tag{4.24}$$

Computing the syndromes according to Equation (4.17),

$$\mathbf{S}_1 = \alpha^8 + \alpha^3 = \alpha^{13} \tag{4.25a}$$

$$\mathbf{S}_2 = \alpha^{16} + \alpha^6 = \alpha^{11} \tag{4.25b}$$

$$\mathbf{S}_3 = \alpha^{24} + \alpha^9 = 0 \tag{4.25c}$$

$$\mathbf{S}_4 = \alpha^{32} + \alpha^{12} = \alpha^7 \tag{4.25d}$$

The determinant, $\det \mathbf{M} = \begin{vmatrix} S_1 & S_2 \\ S_2 & S_3 \end{vmatrix} \neq 0$. Therefore,

$$\mathbf{M} = \begin{bmatrix} \alpha^{13} & \alpha^{11} \\ \alpha^{11} & 0 \end{bmatrix} \tag{4.26}$$

therefore,

$$\mathbf{M}^{-1} = \begin{bmatrix} 0 & \alpha^{4} \\ \alpha^{4} & \alpha^{6} \end{bmatrix} \tag{4.27}$$

and,

$$\begin{bmatrix} \lambda_2 \\ \lambda_1 \end{bmatrix} = \left[\mathbf{M}^{-1}\right] \begin{bmatrix} S_3 \\ S_4 \end{bmatrix} \tag{4.28}$$

$$= \begin{bmatrix} 0 & \alpha^{4} \\ \alpha^{4} & \alpha^{6} \end{bmatrix} \begin{bmatrix} 0 \\ \alpha^{7} \end{bmatrix}$$

$$= \begin{bmatrix} \alpha^{11} \\ \alpha^{13} \end{bmatrix} \tag{4.28a}$$

Thus,

$$\lambda(z) = \alpha^{11} z^2 + \alpha^{13} z + 1 \tag{4.29}$$

$$= (\alpha^{8} z + 1)(\alpha^{3} z + 1) \tag{4.29a}$$

The error locations occur in the 3rd and 8th positions giving $e(X) = X^8 + X^3$. By inverting the digits in these positions of the received vector, error correction is accomplished.

Various numerical methods, e.g., Chien search, Berlekamp's iterative scheme, and the Peterson-Gorenstein-Zierler algorithm, have been developed to assist in solving the error locator polynomial of the BCH codes. Their development and descriptive use may be found in References 9, 10 , and 11.

3. Performance Evaluation

Error performance of BCH codes are based on maximum likelihood decoding principles. Since the weight structure is known only for a small fraction of the family of BCH codes, most decoding algorithms are based on having no knowledge of the code weight structure. This leads to the probability relationship expressed in Equation (3.9).

An upper bound to the decoded bit error probability can be found from Equation (3.8) as modified for BCH coding:

$$P_B \leq \sum_{j=t+1}^{n} \left(\frac{j+t}{n} \right) \binom{n}{j} p^j (1-p)^{n-j} \tag{4.30}$$

Since the values for t, n and k are known for all values up to $n = 1023$, Equation (4.30) can be evaluated as a function of p. For the case of BCH codes, the co-error function is modified to account for code rate, R:

$$p = Q\left(\sqrt{2R \frac{E_s}{N_o}} \right) \tag{4.31}$$

Figure 7 illustrates the upper bound of various BCH codes as derived from Equations (4.30) and (4.31).

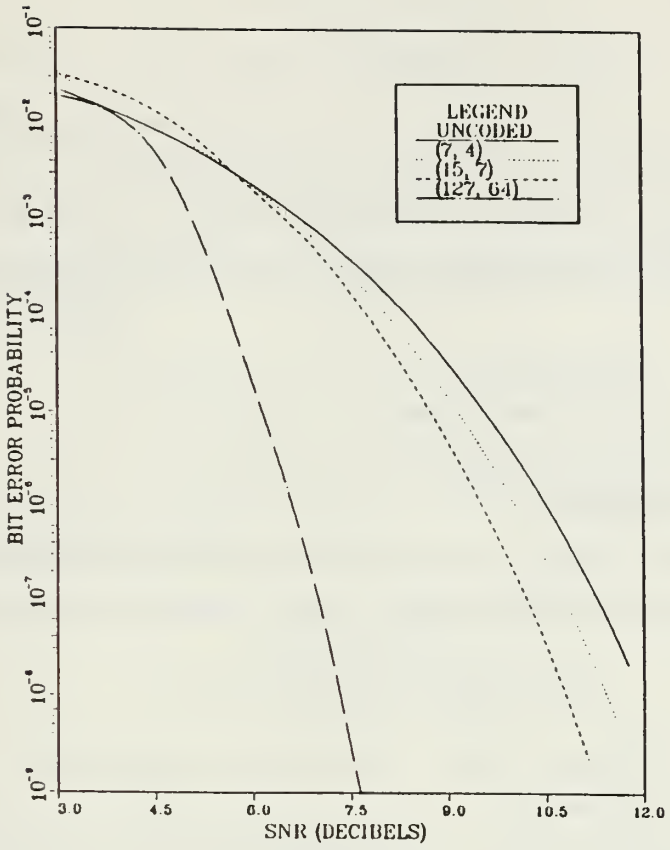

Figure 7. Performance of Various BCH Codes

It has been shown in Reference 12 that between code rates of $1/3 \leq R \leq 3/4$, a broad maximum of coding gain versus code rate for fixed values of n occur. Performance degrades significantly for particularly high or low code rates. Figure 7 illustrates performance of the BCH codes with $R \simeq 1/2$.

E. REED-SOLOMON CODES

A very important subclass of nonbinary BCH codes are the Reed-Solomon (R-S) codes which achieve the largest possible minimum distance (d_{min}) for any linear block code with specified block lengths. For nonbinary codes, the distance between code words is defined as the number of nonbinary symbols by which the two code words differ. The minimum distance for these codes are given by:

$$d_{min} = n - k + 1 \tag{4.32}$$

but since n - k = $2t$,

$$d_{\min} = 2t + 1 \tag{4.33}$$

R-S codes use a non-binary alphabet of 2^m symbols represented by GF(2^m). However, for R-S coding with $m = 1$, $q = 2^m$, the symbol field GF(q) and the error locator field GF(q^m) are the same (as such, R-S codes are examples of a q-ary BCH codes). For $m = 1$, the q-ary BCH code becomes a t-error correcting R-S code with the following parameters:

Block length:	$n = q - 1$
Number of parity digits:	$n - k = 2t$
Minimum distance:	$d = 2t + 1$

1. Generator Polynomial

Using a R-S code with code symbols from the alphabet established by GF(q) where $q = 2^m$, the generator polynomial of a t-error correcting code of length $2^m - 1$ is:

$$g(X) = (X + \alpha)(X + \alpha^2)...(X + \alpha^{2t}) \tag{4.34}$$

where α is a primitve element of GF(q). This is always a polynomial of degree $2t$ and satisfies the $n - k = 2t$ constraint.

As for BCH codes, the R-S code vector is obtained by multiplying the generator matrix and the message vector to produce:

$$\mathbf{c} = \mathbf{mG} \tag{4.35}$$

Let

$$c(X) = c_0 + c_1X + c_2X^2 + ... + c_{n-1}X^{n-1} \tag{4.36}$$

be the transmitted vector and

$$r(X) = r_0 + r_1X + r_2X^2 + ... + r_{n-1}X^{n-1} \tag{4.37}$$

be the received vector. Then the channel error, e(X), is the difference between the transmitted and received vectors and is a symbol from GF(q),

$$e = e_0 + e_1X + e_2X^2 + ... + e_{n-1}X^{n-1} \tag{4.38}$$

or equivalently,

$$\mathbf{e}(X) = \mathbf{r}(X) - \mathbf{c}(X) \tag{4.39}$$

If $\mathbf{e}(X)$ represents an error pattern of less than or equal to t errors, at positions $X^{j_1}, X^{j_2}, X^{j_v}$, then

$$\mathbf{e}(X) = \mathbf{e}_{j_1}X^{j_1} + \mathbf{e}_{j_2}X^{j_2} + \ldots + \mathbf{e}_{j_v}X^{j_v}. \tag{4.40}$$

In order to determine the error vector, the error locations, j_l, and the error values, e_{j_l}, must be found. Defining

$$\beta_l = \alpha^{j_l} \quad \text{for } l = 1, 2, \ldots, v \tag{4.41}$$

as the error locations, the value of the error at the location corresponding to β_l is given by the following equation [Ref. 9]:

$$\mathbf{e}_{j_l} = \frac{\mathbf{Z}(\beta_l^{-1})}{\prod_{\substack{i=1 \\ i \neq l}}^{v} (1 + \beta_i\beta_l^{-1})} \tag{4.42}$$

where

$$\mathbf{Z}(X) = 1 + (S_1 + \sigma_1)X + (S_2 + \sigma_1 S_1 + \sigma_2)X^2 + \ldots$$

$$+ (S_v + \sigma_1 S_{v-1} + \sigma_2 S_{v-2} + \ldots + \sigma_v)X^v \tag{4.43}$$

where $S_1, S_2, \ldots, S_v$ are the syndrome components S_i s.

2. Decoding of R-S Codes

In order to decode the R-S code, the procedure is the same as that for any q-ary BCH code with the addition of a final step. Final calculation of the error values using Equation (4.42) is necessary. The best way to demonstrate R-S decoding is with an example [Ref. 9].

Consider a R-S code with symbols from GF(2^4) and $t = 3$. The generator polynomial is:

$$\mathbf{g}(X) = (X + \alpha)(X + \alpha^2)(X + \alpha^3)(X + \alpha^4)(X + \alpha^5)(X + \alpha^6) \tag{4.44}$$

$$= \alpha^4 + \alpha^{10}X + \alpha^3X^2 + \alpha^9X^3 + \alpha^9x^4 + \alpha^3X^5 + X^6 \tag{4.44a}$$

Assume that the all-zero vector was transmitted and **r**, the received vector is given by **r** = $(000\alpha^7 00\alpha^3 00000\alpha^4 00)$. As such, $\mathbf{r}(X) = \alpha^7 X^3 + \alpha^3 X^6 + \alpha^4 X^{12}$. The syndrome components are generated using Equation (4.17):

$$\mathbf{S}_1 = \mathbf{r}(\alpha) = \alpha^{10} + \alpha^9 + \alpha = \alpha^{12} \tag{4.45a}$$

$$\mathbf{S}_2 = \mathbf{r}(\alpha^2) = \alpha^{13} + 1 \tag{4.45b}$$

$$\mathbf{S}_3 = \mathbf{r}(\alpha^3) = \alpha + \alpha^6 + \alpha^{10} = \alpha^{14} \tag{4.45c}$$

$$\mathbf{S}_4 = \mathbf{r}(\alpha^4) = \alpha^4 + \alpha^{12} + \alpha^7 = \alpha^{10} \tag{4.45d}$$

$$\mathbf{S}_5 = \mathbf{r}(\alpha^5) = \alpha^7 + \alpha^3 + \alpha^4 = 0 \tag{4.45e}$$

$$\mathbf{S}_6 = \mathbf{r}(\alpha^6) = \alpha^{10} + \alpha^9 + \alpha = \alpha^{12} \tag{4.45f}$$

Using an iterative algorithm developed by Berlekamp, the error location polynomial $\sigma(X)$ must be found. The description of this algorithm is a lengthy process which may be found in References 9 and 13. The following is offered without proof. The error location polynomial $\sigma(X) = 1 + \alpha^7 X + \alpha^4 X^2 + \alpha^6 X^3$. By substituting $1, \alpha, \alpha^2, \ldots, \alpha^{14}$ in $\sigma(X)$, the roots of σ (X) are: α^3, α^9 and α^{12}. These are the error location numbers of the error pattern **e**(X) and the errors occur at positions X^3, X^6 and X^{12}.

Now the error values, $\mathbf{e}_{j_l}$, must be computed according to Equation (4.42):

$$\mathbf{e}_3 = \frac{1 + \alpha^2\alpha^{-3} + \alpha^{-6} + \alpha^6\alpha^{-9}}{(1 + \alpha^6\alpha^{-3})(1 + \alpha^{12}\alpha^{-3})} \tag{4.46}$$

$$= \frac{1 + \alpha^{14} + \alpha^9 + \alpha^{12}}{\alpha^{14}\alpha^7} = \frac{\alpha^{13}}{\alpha^6} = \alpha^7$$

$$\mathbf{e}_6 = \frac{1 + \alpha^2\alpha^{-6} + \alpha^{-12} + \alpha^6\alpha^{-18}}{(1 + \alpha^3\alpha^{-6})(1 + \alpha^{12}\alpha^{-6})} \tag{4.47}$$

$$= \frac{1 + \alpha^{11} + \alpha^3 + \alpha^3}{\alpha^9} = \frac{\alpha^{12}}{\alpha^9} = \alpha^3$$

$$\mathbf{e}_{12} = \frac{1 + \alpha^2\alpha^{-12} + \alpha^{-24} + \alpha^6\alpha^{-36}}{(1 + \alpha^3\alpha^{-12})(1 + \alpha^6\alpha^{-12})} \tag{4.48}$$

$$= \frac{1 + \alpha^5 + \alpha^6 + 1}{\alpha^5} = \frac{\alpha^9}{\alpha^5} = \alpha^4$$

Thus the final error pattern is:

$$\mathbf{e}(X) = \alpha^7 X^3 + \alpha^3 X^6 + \alpha^4 X^{12} \tag{4.49}$$

This corresponds to $\mathbf{r}(X) - \mathbf{c}(X)$ so that the final decoded message is found by forming $\mathbf{r}(X) - \mathbf{e}(X)$ and the transmitted all-zero vector is therefore recovered.

3. Performance Evaluation and Application

The greatest advantage of the R-S codes is that they achieve the largest possible code minimum distance, $d_{\min}$. This distance is given by:

$$d_{\min} = n - k + 1 \tag{4.50}$$

Therefore, R-S coding can correct up to t errors as follows:

$$t = \frac{d_{\min} - 1}{2} = \frac{n - k}{2} \tag{4.51}$$

and therefore only requires $2t$ parity bits.

In binary (n, k) coding, the entire n-tuple space is comprised of 2^n binary words. Similarly, the k-tuple space is made up of 2^k binary words. Since R-S codes are nonbinary, the n-tuple space is expanded to nm-tuple where m is the number of bits used to represent each symbol. This expands the codeword space to 2^{mn}, a number significantly larger than 2^n. This also makes the km-tuple space larger but the ratio of k/m decreases. The result is that when small fractions of n-tuple space are used for codewords, a code with large $d_{\min}$ is created.

The decoded bit error probability associated with a R-S code follows from the results shown in Equation (3.8), with a slight modification. The upper bound to P_B is given by

$$P_B \leq \frac{2^{m-1}}{2^m - 1} \sum_{j=t+1}^{n} \left(\frac{j + t}{n} \right) \binom{n}{j} p^j (1 - p)^{n-j} \tag{4.52}$$

where the scaling factor $\frac{2^{m-1}}{2^m - 1}$ accounts for the average number of information bits per symbol error.

F. CONVOLUTIONAL CODES

Previous sections have presented various block codes where the data is grouped in k-bit words and coded into n-bit words where n/k represents the redundancy necessary for error detection and correction. Contrasting this technique are convolutional codes where the encoder accepts data in groups of k symbols and provides an encoded output in groups of n symbols, where $n > k$. However the encoder output not only depends on the current input data but also on previous data blocks; that is, the encoder has memory. Convolutional coding is affected by constraint length (number of stages in the shift register) and code rate, $R = k/n$. Significant coding gains and the availibility of decoding algorithms which can take advantage of soft receiver decisions, and the relative ease with which both the encoder and decoder can be implemented have lead to widespread use of convolutional codes.

The most commonly used convolutional codes utilize $k = 1$. In this case, information bits are shifted into the encoder one bit at a time. As the i^{th} bit is applied to the input of the shift register, all previous bits are sequentially shifted one stage to the right and modulo-2 summed as prescribed by the generating polynomial. The output is then transmitted while the $i^{th} + 1$ bit is applied to the shift register input and the procedure repeated.

Examination of convolutional coding involves characterization of the encoder and decoder. The encoder can be described by one of three approaches, namely: 1) the polynomial matrix approach, 2) the scalar matrix approach, and 3) the shift register approach. In this discussion, the later will be used. Likewise, decoder characterization can also be described through one of three approaches: 1) state-diagram approach, 2) tree approach, and 3) trellis approach.

1. Shift Register Approach

A (1/2,3) convolutional encoder with generator polynominals $(X^2 + X + 1)$ and $(X^2 + 1)$ shown in Figure 6, will be used as an example. Selection of generator polynomials is purely arbitrary and is limited only by the number of stages in the shift register. Let the message vector $\mathbf{m} = (1\ 0\ 1)$, the shift register initially loaded with all zeros and the output branch word formed as $u_1\ u_2$.

The initial message input generates an output of 1 1 from the modulo-2 adders. As the next message bit is applied, the initial bit is shifted one stage to the right and the output is now 1 0. This procedure continues until the message vector has passed through the shift register and $(K - 1)$ zeros are input in order to flush the register to its initial all zero state. In this example the ouput sequence is: 11 10 00 10 11.

Any number of shift registers and modulo-2 adders may be used depending on the complexity desired. Generating polynomials will determine the number and location of modulo-2 adders.

2. Trellis Decoding Approach

The trellis diagram for the (1/2, 3) convolutional encoder is shown in Figure 8.

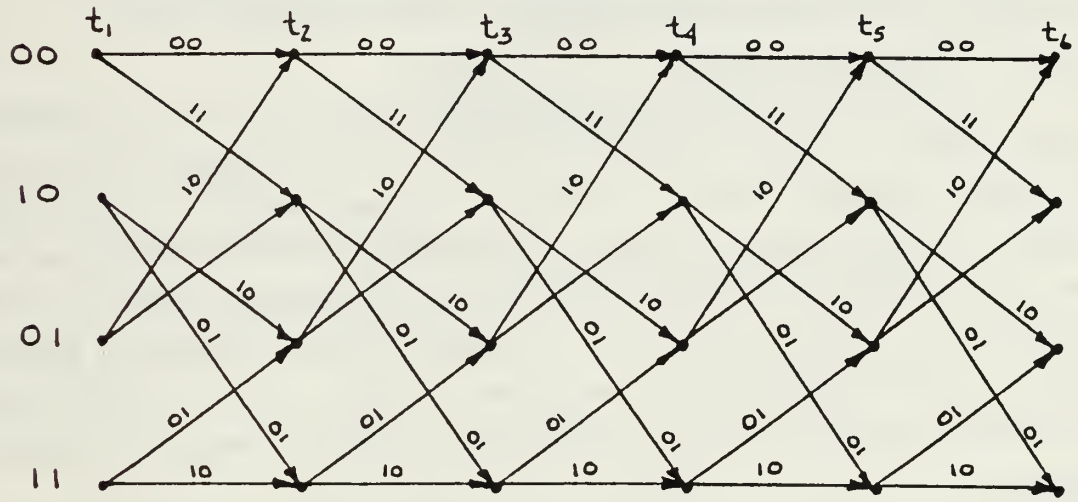

Figure 8. Encoder Trellis Diagram: (Rate 1/2, $K = 3$)

In order to decode using a trellis diagram, the trellis encoder must first be generated. It is formed as follows: from any time state, t_i where $i = 1, 2, ..., 2K$, there are one of two transitions that may occur; these represent either a 1 or 0 being applied to the shift register input. By mapping the result of each time period to the next time period, t_{i+1}, the trellis can be completed. The symbols on each of the branches of the trellis are the modulo-2 added output as a result of transitions from one time period to the next. For instance, referring to Figure 8, at $t = 3$ in the 1 0 state an input of 0 will result in a transition to the 0 1 state at t_4 along the branch labeled 1 0. The output as a result of this transition would be the same as the branch label, 1 0. An input of 1 will result in transition to the 1 1 state at t_4 along the branch labeled 0 1. A completed encode trellis diagram represents all possible state-time transitions and is fundamental to trellis decoding.

The decoding algorithms of trellis codes can be lengthy and complicated processes. As such, these will not be explained in this work. References 13 and 14 are recommended for detailed discussions of trellis decoding. The primary objective of trellis decoding is to find the singular path through all time states with the least path metric. Evaluating the necessary steps to get from t_i to t_{i+1} along the least path metric will result in the recovery of the transmitted message with the fewest, possibly zero errors.

Viterbi's convolutional decoding algorithm [Ref. 15] made use of trellis decoding combined with maximum likelihood decisions. This allows the option of soft-decision decoding,[4] resulting in recovered message with a lower probability of error.

3. Performance Evaluation

Performance evaluation for convolutionally encoded data is not as straightforward a procedure as with both BCH and R-S coding. As stated earlier, there are three approaches one may take when describing convolutional coding. In describing performance evaluation, most approaches use a combination of all three descriptive methods. In this work, only one approach was used in detailing convolutional codes. Therefore, some terms, associated with the other two approaches, must be defined in order to continue the derivation of performance equations. Most terms will be introduced with a minimum of discussion. Reference 15 provides the necessary detailed derivations.

The first term that must be defined is the minimum free distance, d_{free} . This is the weight of the minimum-weight path which begins and ends in the zero state (weight corresponds to the number of 1's in a sequence of binary bits). Its usual derivation comes from the state-diagram approach but an analogy to the trellis approach may be made. In this respect, d_{free} is the minimum path weight a code may have when diverging and remerging to the all-zero path. With this definition in hand, the first event error probability may be expressed:

$$P_f < \sum_{d=d_{free}}^{\infty} n_d P_d \tag{4.53}$$

where n_d is the number of codewords with weight d and P_d is event error probability of weight d. A probability of bit error may be obtained from this by weighting each term

---

[4] Soft decision decoding allows for confidence levels when making bit decisions. These levels are quantization levels in the demodulator and result in more accurate decisions at the cost of demodulator and decoding complexity.

by the information weight for each path (i.e., the number of bit errors). However, with a code rate of R = k/n, there are k symbols decoded on each branch. So now P_B is bounded by:

$$P_B \leq \frac{1}{k} \sum_{d=d_{free}}^{\infty} w_d P_d \tag{4.54}$$

where w_d is the total number of non-zero information bits on all d weight paths.

Now, using the matrix approach, it can be shown from Reference 14 that:

$$P_B \leq \left. \frac{\partial T(D, N)}{\partial N} \right|_{N=1,\, D=2\sqrt{p(1-p)}} \tag{4.55}$$

where $p = Q\left(\sqrt{E_s/N_o}\right)$ and

D ≡ redundancy of the code

N ≡ unicity distance (branch traversed caused by an input of 1)

T ≡ generating function of a code.

From

$$T(D, N) = \sum_{d=d_{free}}^{\infty} \sum_{b=1}^{\infty} n_d D^d N^b \tag{4.56}$$

it follows:

$$\frac{\partial T(D, N)}{\partial N} = \sum_{d=d_{free}}^{\infty} \sum_{b=1}^{\infty} b\, n_d N^{-d} \tag{4.57}$$

$$= \sum_{d=d_{free}}^{\infty} B_d N_d \tag{4.57a}$$

where

$$B_d = \sum_{b=1}^{\infty} b\, n_d \tag{4.58}$$

Now

$$P_B \leq \frac{1}{k} \sum_{d=d_{free}}^{\infty} B_d \left[2\sqrt{p(1-p)} \right]^d \tag{4.59}$$

for a convolutional code with a generating function T(D, N).

For small values of p, Equation (4.59) is dominated by the first term, so that:

$$P_B \approx \frac{1}{k} B_{d_{free}} \left[2\sqrt{p(1-p)} \right]^{d_{free}} \tag{4.60}$$

$$\approx \frac{1}{k} B_{d_{free}} 2^{d_{free}} \, p^{\frac{d_{free}}{2}} \tag{4.60a}$$

This approximation will be used to evaluate the performance of convolutionally encoded data for various codes, and contrasted with the performance of block encoded data.

V. CODING FOR THE MFQPSK SIGNAL

When measuring the benefits of one particular coding scheme versus another, a common reference must be established. All other variables being equal, the coding gain provided by each coding scheme will be that reference. This is the measure of the decrease in the SNR allowed through coding over an uncoded signal at a specific bit error probability. In this evaluation, a bit error probability of 10^{-5} will be the fixed evaluation level and the performance of various coding schemes measured at that point. The corresponding SNR will be compared to the required SNR of the uncoded received signal to achieve the same rate and the corresponding coding gain will be computed. This coding gain will be the final performance measurement for the various coding schemes. This evaluation assumes that all other variables are equal, even though, in practice, this may not be valid assumption. Some codes yield better results in a burst error environment while others are tailored more toward correcting random errors. The complexity of various codes and their ease of implementation also vary. But these factors cannot be measured quantitatively and be objectively weighted in an overall performance evaluation. However, the effects of each of these factors will be discussed so as to at least provide a subjective evaluation of each.

A. BLOCK CODE EVALUATION

The MFQPSK signaling system can be implemented using various packet combinations of M tones and N bauds. A baseline system has been established with M = 64. Encoding phase information at each frequency bin will result in 128 binary bits of information to be encoded in each baud.

1. Hamming Codes

The simplest method for block encoding binary data uses the (7, 4) Hamming code. In the case of a baud length of 128 bits, it is obvious that the bauds must be subdivided into what will be refered to as Hamming blocks. Each Hamming block will contain four bits of information and the requisite three parity bits. In order to use this scheme, 32 Hamming blocks are necessary to encode the required 128 information bits of each baud. Since each frequency bin contains a di-bit describing a phase, coding involves effectively two frequency bins per Hamming block.

The Hamming block method provides a very good code rate at 4/7 (.5714) but is limited by its error single correcting capability. No amount of increased redundancy can improve this constraint.

Another Hamming code that would be useful in the MFQPSK signaling system is the (15, 11) Hamming code. This has a code rate of 11/15 (.7333) but maintains the single error correcting capability. The (15, 11) Hamming code would be effective for block lengths of 16 where the additional bit can be used as a parity check thus enabling the (15, 11) code to detect the existence of a double error or it can be used as a timing or synchronization bit. As shown in Figure 9, the (15, 11) Hamming code provides a coding gain of 1.3 dB at a bit error probability of 10^{-5}.

The use of Hamming codes suggest short block lengths. This is because of the Hamming codes' limit of $t = 1$ error correcting capability. If this limit were applied to the entire baud, then the baud would have an error correcting capability equal to exactly the number of blocks of 16 coded bits. But the errors would have to occur at a rate of no more than one error per block coded bits for the Hamming code to be successful. For this reason, Hamming codes are not useful in environments where burst errors are dominant.

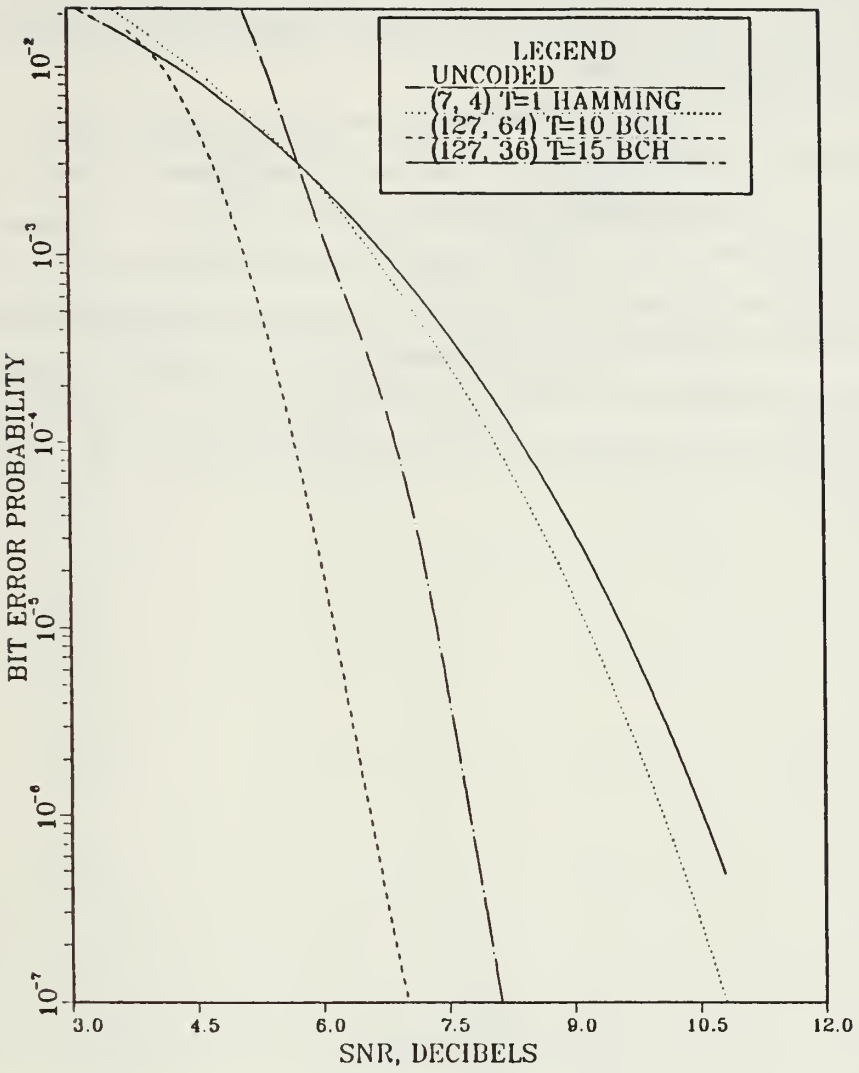

Figure 9. Bit Error Probability for Several Block Codes

2. BCH Codes

Lin and Costello [Ref. 13, Appendix C] have documented BCH codes up to length $2^{10} - 1$, showing various combinations of block lengths and information bits along with their respective generating polynomials. While creating an inordinately large block length, like (1023, 11) $t = 255$, in order to take advantage of the large error correcting capability, the added block length increases the transmission bandwidth and the de-

creases code rate (.011 in this example). This is unacceptable for the application of interest here. To combat this, it is desired to have the block length long enough to encode the data with a code that has an acceptable error correcting capability. In the example of transmitting 64 frequencies (128 information bits) two approaches may be taken.

a. A (127, 64) BCH Code

To maintain a baud-to-block length continuity, a (127, 64) with $t = 10$ BCH code can be used. In this case, each baud that is transmitted contains 64 information bits as well as the 63 parity bits in order to generate a block of 127 coded bits. A block parity bit may be added to increase block length to 128 elements. Computer programs used to evaluate the performance of various schemes utilizing BCH codes (see Appendix A) have shown that the effect of the added parity bit is negligible. That is, the SNR at any bit error probability is decreased by less than .1 dB. At a bit error probability of 10^{-5}, the coding gain achieved for the (127, 64) BCH code is 2.8 dB. This result is shown in Figure 10.

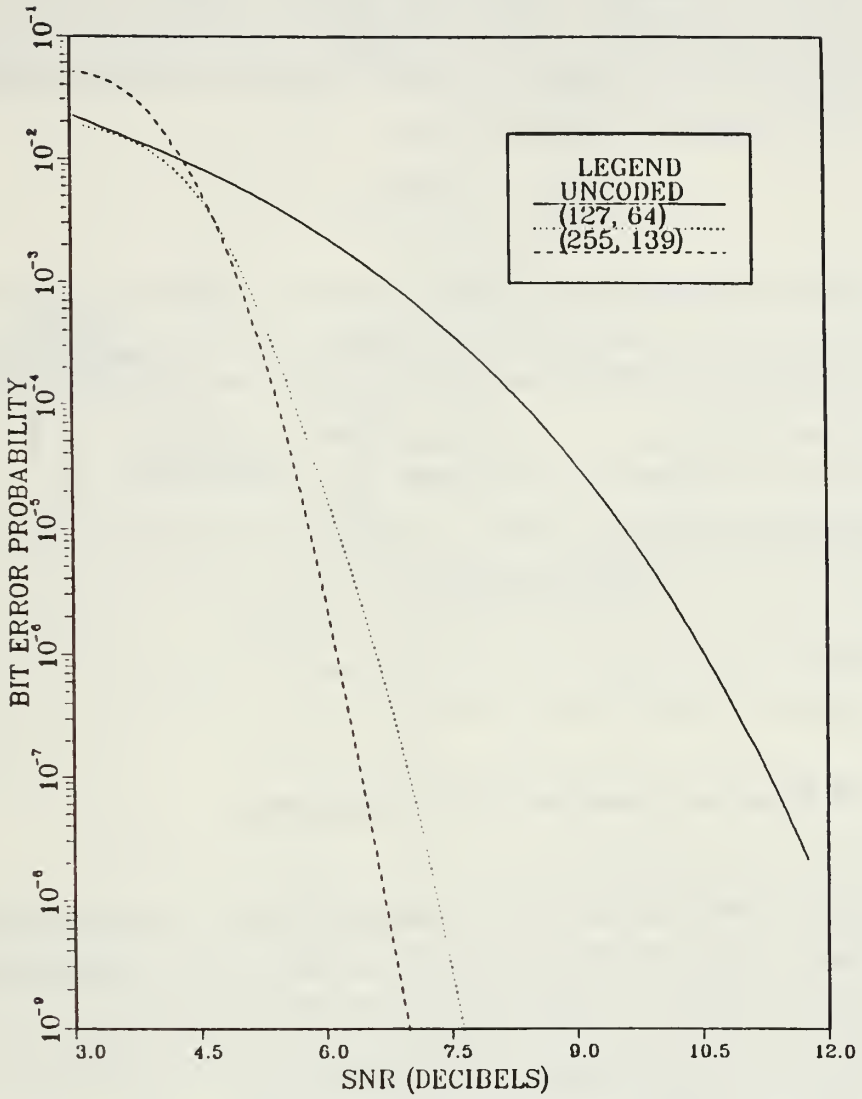

Figure 10. Bit Error Probabilities for BCH Codes

b. A (255, 139) BCH Code

In order to include all 128 information bits of one baud in a single block, a (255, 139) with $t = 15$ BCH may be used. This will maintain baud continuity in that all the 128 information bits are encoded in the same block. The (255, 139) code allows for 139 information bits, 11 more than that contained in the individual baud. What to do with the extra information positions can be the subject of further research. Possibil-

ities include using the 11 extra information positions as synchronization data bits. In this manner, the signal will be synchronized every baud and eliminates the necessity of a synchronization tone or preamble. Perhaps cryptographic keying information may be contained in the 11 extra positions. Again the addition of an overall parity bit, like the previous (127, 64) BCH code is necessary. Figure 11 shows the information/parity bit relationship of the modified (255, 139) BCH code.

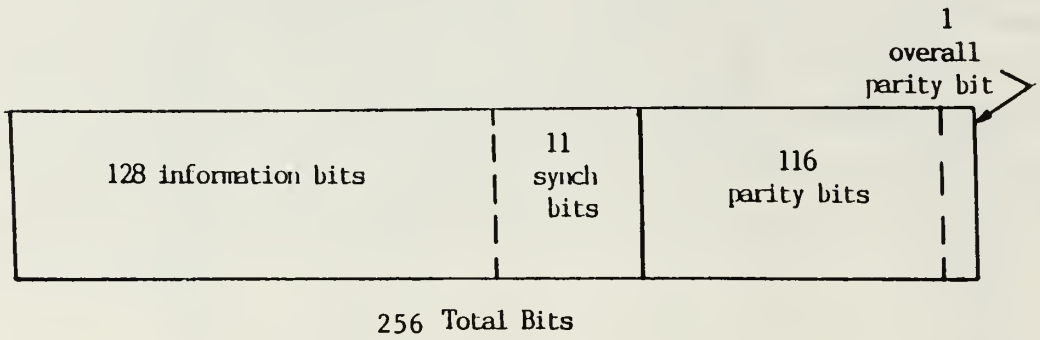

Figure 11. Bit Stream for a Modified (255,139) BCH Code

Figure 10 shows that at a bit error probability of 10^{-5}, the coding gain of the (255, 139) BCH code is 3.8 dB. This is a substantial decrease in SNR at a given bit error probability and clearly illustrates the advantages of error correcting codes. It should also be added that increased redundancy does not always mean improvements in error performance and, hence, coding gain. Performance degrades substantially at very high and very low data rates. For medium code rate block codes (1/3 to 1/4), a broad range of coding gains occur and are subject to increased redundancy. Outside this region, redundancy does little to increase coding gains.

BCH codes, because they are a generalized version of Hamming codes, are more effective as random error correcting codes and have limited use in a burst error environment. However, BCH codes do offer the advantage of variable block lengths and improved performance with low implementational complexity.

3. Reed-Solomon Codes

Reed-Solomon codes offer significant error correcting capabilities. This is because they have the largest possible code minimum distance of any block code of the same length. Minimum distance equals $n - k + 1$ and the t-error correcting capability is:

$$t = \frac{d_{\min} - 1}{2} = \frac{n - k}{2}$$

Because of their large error correcting capability and corresponding moderate code rates, these codes tend to offer great coding gains. Continuing example in which processing 128 binary information bits is required, two particular R-S are considered.

a. A (127, 65) R-S Code

Using arguments similar to those used previously in the discussion of the (127, 64) BCH code, the (127, 65) with $t = 31$ R-S code can maintain the baud-to-block length continuity. However, an information sequence parity check bit is needed for the code's completion. An additional overall parity check bit will result in a block length of 2^l where l is an integer. The error performance associated with this code is illustrated in Figure 12. At a bit error probability of 10^{-5}, the (127, 65) R-S code has a coding gain of 7.5 dB.

b. A (255, 133) R-S Code

The (255, 133) with $t = 61$ R-S code offers large error correcting capability and coding gains. In the application of processing 128 information bits, information bit continuity is maintained per block. That is, all 128 information bits are contained in one encoded block. But again, like its BCH counterpart, signal bandwidth is doubled because the encoded block is approximately twice as long as the information bits contained in the baud. The extra information bits available in the block code (five in this case) and the addition of an overall parity check bit occur here in a manner similar to that encountered for BCH codes.

The greatest advantage of Reed-Solomon coding is the ability to correct errors that occur in short bursts where the bursts approximately match the symbol size, m, in a 2^m length code. Conversely, R-S codes do not perform as well as BCH codes in a random error environment.

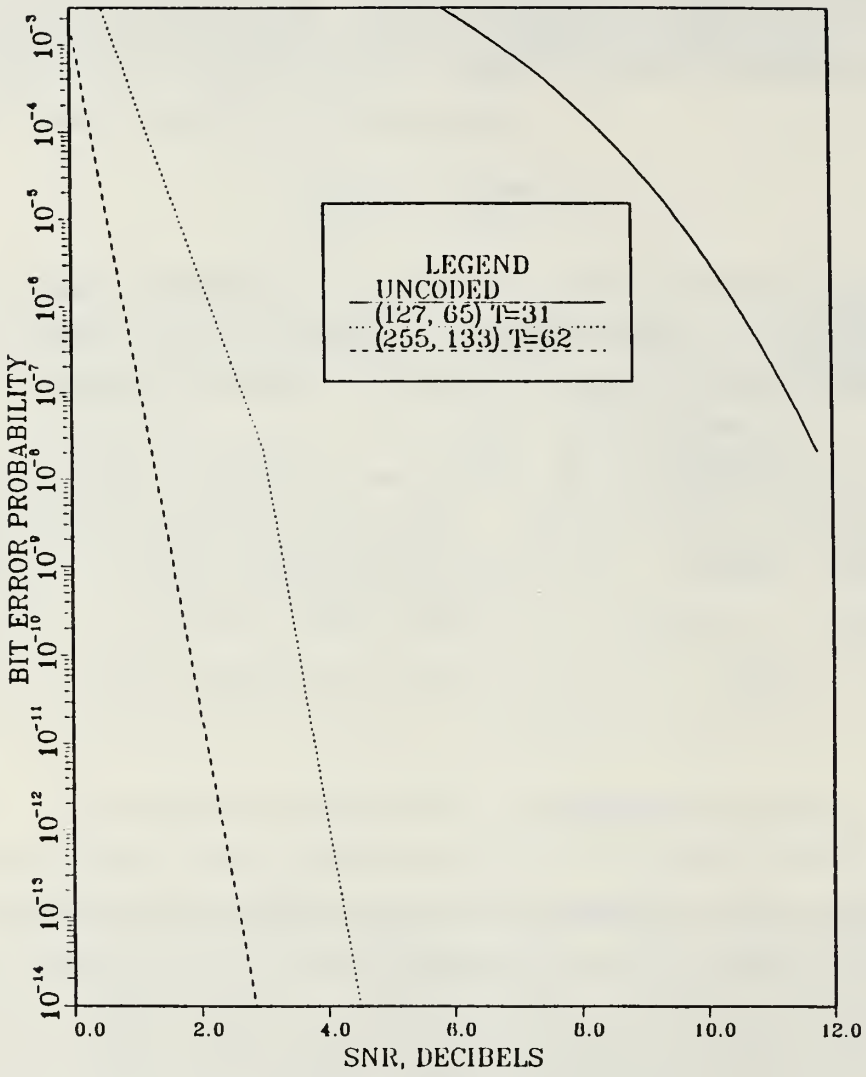

Figure 12. Bit Error Probabilities for Reed-Solomon Codes

B. CONVOLUTIONAL CODE EVALUATION

From Equation (4.60a) the variables involved in specifying the performance of a convolutional code are primarily influenced by the values of d_{free}. This factor, in turn, is a parameter of the constraint length of a code, K. It is desirable, in terms of performance levels, to use the largest value of K without unnecessarily complicating the encoder/decoder structure. Odenwalder investigated a code's constraint length versus

coding complexity and documented the results in Ref. 16. A convolutional code of constraint length, $K = 7$, using a Viterbi decoding algorithm was found to yield the best overall performance characteristics. Therfore only convolutional codes with $K = 7$ and Viterbi decoding will be considered here. An additional condition that must be established prior to the code's evaluation is the code rate. While evaluating possible block codes in the previous section, code rates very near 1/2 were considered. In order to preserve a consistent evaluation, convolutional codes with $R = 1/2$ will be evaluated. Code rates of less than 1/2 will provide better decoded bit error rate performance but this improved performance will come at the cost of bandwidth and receiver complexity.

With the above conditions ($K = 7$, $R = 1/2$), convolutional codes can be considered for which the free distance, d_{free} is a fixed value of 10. At $d_{free} = 10$, $B_{d_{free}} = 36$. (These values are tabulated for various values of d_{free} on page 625 of Reference 15.) A computer program (see Appendix B) was used to evaluate various convolutional codes based on combinations of d_{free} and $B_{d_{free}}$. The results are illustrated in Figure 13. The coding gain achieved can be measured from Figure 13 as 3.2 dB for $d_{free} = 10$. Increasing d_{free} to 12 generates a $B_{d_{free}}$ of 211 and a coding gain of 3.9 dB. The free distance can be increased as the Hamming distance between two codewords is increased. However, this will result in a different value for $B_{d_{free}}$ as tabulated in Reference 15.

To complete the examples presented Figure 13, a $K = 5$, $R = 1/2$ convolutional code is included. In this case, $d_{free} = 7$ and $B_{d_{free}} = 4$. This example is included because a $K = 5$, $R = 1/2$ code is available in a single integrated circuit and would be simple to implement. Using the convolutional code incorporated in this integrated circuit, a coding gain of 2.5 dB will be achieved.

Convolutional codes in general respond very well in a burst error environment. Specific classes of convolutional codes have been developed to further improve the burst errors correcting capabilities. The Berlekamp-Preparata code and the Iwadare-Massey code are two of these codes and are discussed on pages 430 - 440 of Reference 14.

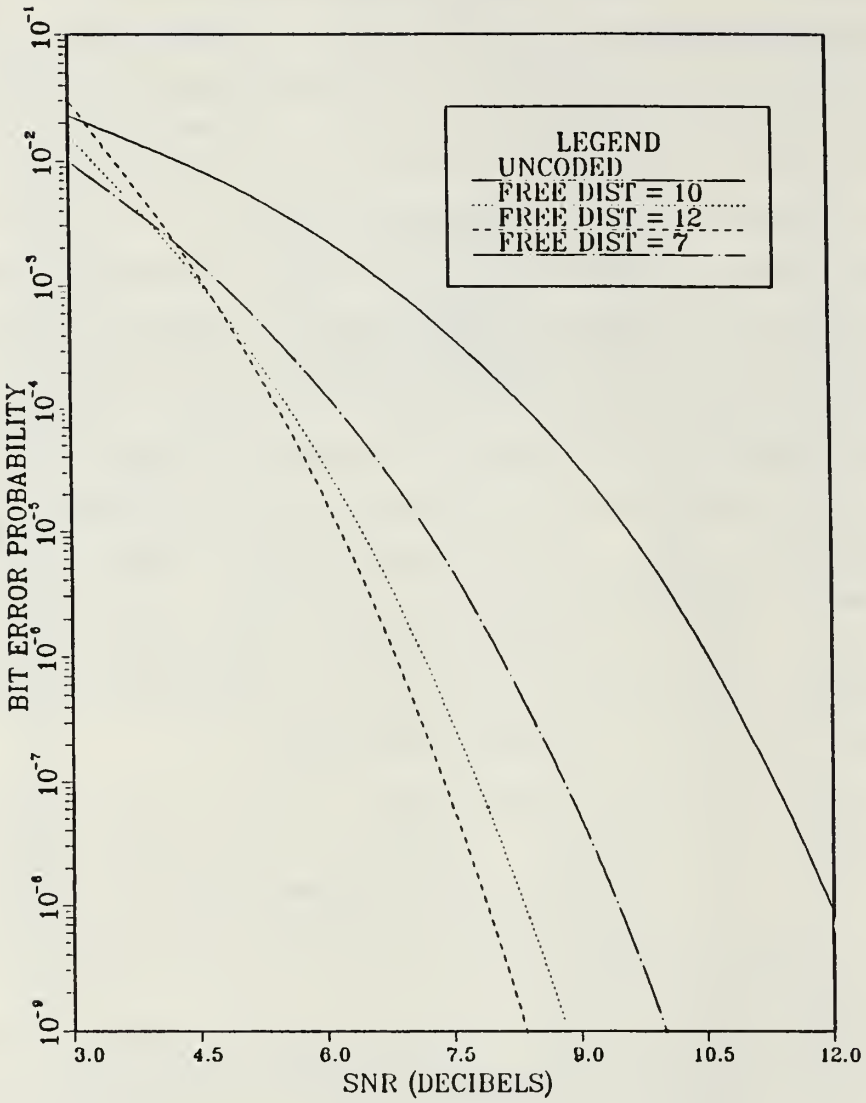

Figure 13. Bit Error Probabilities for Various Convolutional Codes

VI. CONCLUSIONS

A. THE MFQPSK SIGNALING SYSTEM

This study has presented the fundamental principles of a multi-frequency modulation (MFM) scheme. The major factor for considering this scheme is the potential use of this modulation technique in a variety of media. More specifically, the MFM scheme was directed towards a Multiple Frequency Quaternary Phase Shift Keyed (MFQPSK) signaling system in an underwater acoustic signaling application. The MFQPSK signaling system was first presented, and then followed by the establishment that error detection/correction (EDC) is a desireable feature for insuring fidelity of the received signal. However with the many EDC coding schemes available, a determination of those best suited for the signaling system under consideration had to be made. Decoded bit error rate performance using various block and convolutional codes were evaluated and a computer program generated to calculate and plot these bit error probabilities as a function of SNR. After considering specific coding schemes (such as Hamming, Golay, BCH, R-S, and convolutional codes), a determination was made as to how each of the selected coding schemes could be used in conjunction with the MFQPSK signaling system.

The performance of each of the many block codes and convolutional codes vary when subjected to different channel environments. In an environment in which random errors most often occur, Hamming, Golay, and BCH codes outperform R-S and convolutional codes as a whole. This is a broad generalization, but one which can be substantiated by evaluating error performance associated with each of the codes. It is evident that if the channel environment can be successfully predicted (determination of burst versus random error conditions), then the EDC coding selection would be simplified. However, the channel environment cannot be accurately predicted. Furthermore, the channel medium must be characterized and modeled accurately. However the degree of complexity of this effort is such that a separate project is being carried out, concurrently with this study, that involves modeling an underwater acoustic channel as the medium for propogation of the MFQPSK signal. Until such time as experimental results can verify the channel model being developed, all results are subject to the validity of the assumptions in the model.

When using block codes for EDC in the MFQPSK system, selection of baud length to be transmitted will affect the selection of a block code to be used. Hamming codes provide adequate coding gains when used as a short block code in a random error environment, but are limited by their error correcting capability. BCH codes yield good coding gains at the specified value of bit error probability used in this project. BCH codes offer a wide selection of block lengths and error correcting capabilities, with good responcs when used in environments where random errors dominate. R-S codes provide substantial coding gains, however at a price that R-S codes being q-ary codes, block lengths are large (a R-S code where $q = 2$ is the same as a binary BCH code). These large block lengths add complexity to the encoder and decoder structures.

A class of codes that perform well in environments where either type of error, random or burst, occur are the group of convolutional codes. Since convolutional code's performance is dependent on the number of stages in the shift register used to generate the code, their implementation tends to be less complex than that of block codes. The number of stages in the shift register, called the constraint length, affects two parameters, namely (d_{free} and $B_{d_{free}}$) which are used in the evaluation of bit error rate performance of convolutionally encoded data. Once a constraint length is established, these two parameters can be determined and performance evaluated. An advantage of applying convolutional codes to the MFQPSK signal system is the inherent ability to process the bit stream continuously. As bauds become filled with information (binary data), they can be processed through the convolutional encoder while other bauds are filled with more information.

In conclusion, convolutional codes were found to provide adequate values of bit error probabilities over a wide range of SNR's and significant coding gains. Additionally, they are relatively easy to implement and are well suited to the MFQPSK signaling system and its signal structure.

B. AREAS OF CONTINUED RESEARCH

The EDC codes studied in this project were developed singularly. Coding schemes used in existing communication systems use a combination of these individual codes. Such codes are referred to as concatenated or interleaved codes. Concatenation is a method of constructing long codes from shorter codes. Concatenated coding can be explained by the following: encode an information bit stream initially using a R-S code and then process the resulting encoded bit stream through a convolutional encoder. This double coding results in concatenation. Coding gains provided by concatenated codes

and their application to the MFQPSK signaling system present research areas which have as yet not been addressed. Additionally, as channel models are further refined, various EDC coding schemes that incorporate these channel models can be further refined, analyzed, and verified by experimental results.

APPENDIX A. BCH AND R-S CODING PERFORMANCE EVALUATION

The purpose of this computer program is to evaluate and plot BCH and R-S coding performance curves on the IBM mainframe computer at the Naval Postgraduate School with calls to the IMSL10 library. The output graph will be a semi-logarithmic scale of Bit Error Probability vs. SNR. The input data will consist of the code's description. For example, if a (127, 64) $t = 10$ BCH code were to be evaluated, the inputs would be $K = 64.0$, $n = 127$, and $t = 10$. It is imperative that the values are used as described in the initial declarations. Specifically, n and t must be integers. This permits successful calls to subroutines and functions external to this program. The equations this program is designed to calculate are given by Equations (4.30) and (4.31) for BCH codes, and by Equation (4.52) for R-S codes.

Flexibility is built into this program so that R-S codes may also be computed using this program. Program lines which need modification depending on which code is being computed are noted with an asterisk (*).

```
C
C
C
C     VARIABLE DECLARATIONS
C
      REAL BC, JJ, TT, NN, Z, SF
      REAL R, SNR, XPLT(2000), YPLT(2000), S, TWO, K
      DOUBLE PRECISION X, Q, P1, P2
      DOUBLE PRECISION PROB, TEMP1, TEMP2
      INTEGER II, J, T, N, TP1
C
C
C     VARIABLE INPUTS
C
      K = 64.0
      N = 127
      T = 10
C  *  M = 7
C     This is needed to generate the scaling factor for R-S codes and
C     is different for each code.  M=7 is an example only.
C
C     INITIALIZATION
C
      II = 0
      R = K/N
      J = T + 1
      TP1 = J
      TWO = 2
      S = SQRT(TWO)
C  *  SF = (2 ** (M - 1))/((2 ** M) -1)
```

```
C     This is the scaling factor for R-S codes.
C
C     PROGRAM COMPUTATION.  This starts the loop.  For each value
C     of Z, a Q(.) is calculated and later used in computing
C     probability.  The DERFC is a double precision intrinsic
C     function describing the co-error function as defined in
C     Equation (3.2).  The BINOM is another intrinsic function
C     which computes the binomial coefficient of N and J for
C     each iteration of J.  R represents the code rate, K/N.
C
      DO 100, Z = 1, 40, 1
         X =  SQRT(2 * R * Z)
         Q = .5 * DERFC(X/S)
         SNR = 10 * ALOG10(Z)
         PROB = 0.0
         DO 80, J = TP1, N
            BC = BINOM(N,J)
            NN = N
            JJ = J
            TT = T
            P1 = ((JJ + TT)/NN)
            P2 = ((1 - Q) ** NN)
            TEMP1 = P1 * BC * P2
            TEMP2 = (Q/(1-Q)) ** (JJ/100.0)
                DO 20, K = 1, 100
                   IF (TEMP1 .GE. 1E-75) THEN
C
C                     This IF statement was necessary to prevent
C                     underflow problems on the IBM mainframe.
C                     Any value less than 1E-75 was considered
C                     equal to zero.  While this is recognized as
C                     undesirable, it was necessary to continue with
C                     the program.
C
                     TEMP1 = TEMP1 * TEMP2
                     ELSE
                     TEMP1 = 0.0
                     ENDIF
  20             CONTINUE
C  *           IF BCH THEN
               PROB = PROB + TEMP1
C  *           ELSE, R-S
C  *           RS = SF * PROB
C
               WRITE (81,200) SNR, PROB
  80        CONTINUE

               II = II + 1
               XPLT(II) = SNR
               YPLT(II) = PROB

C

  100 CONTINUE
C
```

```
  200 FORMAT(2X, F7.4, 2X, D12.7,/)
C
C
C     PLOTTING. This plotting routine is unique to the DISSPLA graphics
C     program, version 9.0.  Explanations and applications are best
C     addressed by the reference manual and will not be individually
C     considered.
C
      CALL COMPRS
      CALL PAGE(11,8.5)
      CALL NOBRDR
      CALL AREA2D(6,8)
      CALL XNAME(' SNR $',100)
      CALL YNAME(' BIT ERROR PROBABILITY $',100)
      CALL HEADIN(' BCH CODED PROBABILITY $',100,4,1)
      CALL GRACE(0.0)
      CALL YLOG(4.0,1.5,0.0000000001,1.0)
      CALL SETCLR('MAGENTA')
      CALL CURVE(XPLT,YPLT,10,0)
      CALL ENDGR(1)
      CALL ENDPL(1)
      CALL DONEPL
C
      STOP
      END
```

APPENDIX B. CONVOLUTIONAL CODING PERFORMANCE EVALUATION

The purpose of this algorithm is to compute and plot convolutional coding performance curves. Equation (4.60a) defines the equation used in this program. As in Appendix A, the semi-logarithmic axes will be Bit Error Probability and SNR.

Variables are defined as follows:

$$BDF \Rightarrow B_{d_{free}}$$

and

$$DF \Rightarrow d_{free}$$

and $K \Rightarrow$ constraint length of the code.

```
C
C
C     VARIABLE DECLARATIONS
C
      REAL DF, BDF, K, S, SNR, XPLT(500), YPLT(500), T, Z
      DOUBLE PRECISION X, Q, PROB
      INTEGER II
C
C
C     VARIABLE INPUTS
C
      DF = 7.0
      BDF = 4.0
      K = 7.0
C
C
C     INITIALIZATION
C
      D2 = DF/2
      T = 2.0
      S = SQRT(T)
      II = 0
C
C
C     PROGRAM COMPUTATION.  For each value of Z, a Q(.) is determined.
C     DERFC is a double precision intrinsic function describing the
C     co-error function  as defined in Equation (3.2).
C
      DO 100, Z = 1, 20, 1
         X = SQRT(Z)
         Q = .5 * DERFC(X/S)
         SNR = 10 * ALOG10(Z)
```

```
         P = (1/K) * BDF * (2 ** DF) * (Q ** D2)
C
         WRITE (9, 200) SNR, PROB
         II = II + 1
         XPLT(II) = SNR
         YPLT(II) = PROB
 100  CONTINUE
C
 200  FORMAT (2X, F7.4, 2X, D12.7,/)
C
C     PLOTTING.  This plotting routine is unique to the DISSPLA graphics
C     program, version 9.0.  Explanations and applications are best
C     addressed by the reference manual and will not be individually
C     considered.
C
      CALL COMPRS
      CALL PAGE(11,8.5)
      CALL NOBRDR
      CALL AREA2D(6,8)
      CALL XNAME( 'SNR $',100)
      CALL YNAME( 'PROBABILITY $',100)
      CALL HEADIN( 'CONVOLUTIONAL CODE PERFORMANCE $',100,4,1)
      CALL GRACE(0.0)
      CALL YLOG(3.0,1.5,0.000000001,1.0)
      CALL SETCLR('MAGENTA')
      CALL CURVE(XPLT,YPLT,20,0)
      CALL ENDGR(1)
      CALL ENDPL(1)
      CALL DONEPL
C
      STOP
      END
```

LIST OF REFERENCES

1. Moose, P. H., "Theory of Multi-Frequency Modulation (MFM) Digital Communications," Technical Report No. NPS62-89-019, Naval Postgraduate School, Monterey Ca., May 1989.

2. Moose, P. H., "Acoustic Tactical Data Link", **Proceedings of MILCOM 86,** Monterey, Ca., October 1986.

3. Lambert, J. D., **Initial Design and Feasibility Analysis of ATDL,** Master's Thesis, Naval Postgraduate School, Monterey, Ca., June, 1984.

4. Childs, R. D., **High Speed Output Interface for a MFQPSK Signal Generated on an Industry Standard Computer.** Master's Thesis, Naval Postgraduate School, Monterey, Ca., December, 1988.

5. Bukofzer, D., Notes for EC4550 (Digital Communications), Naval Postgraduate School, 1989 (unpublished).

6. Sklar, B., **Digital Communincations, Fundamentals and Applications** , Prentice Hall, 1988.

7. Pless, V., **Introduction to the Theory of Error-Correcting Codes,** John Wiley and Sons, 1982.

8. Hamming, R. W., "Error Detecting and Error Correcting Codes," **Bell Systems Technical Journal,** v. 29, April 1950.

9. Lin, S., **An Introduction to Error-Correcting Codes,** Prentice-Hall, 1970.

10. Blahut, R. E., **Theory and Practice of Error Control Codes,** Addison-Wesley Publishing Company, 1983.

11. Clark, G. C. and Cain, J. B., **Error -Correction Coding for Digital Communications**, Plenum Press, 1981.

12. Wozencraft, J. M. and Jacobs, I. M., **Principles of Communication Engineering**, John Wiley and Sons, 1965.

13. Lin, S. and Costello, D. J., **Error Control Coding**, Prentice-Hall, 1980.

14. Ziemer R. E. and Peterson, R. L., **Digital Communications and Spread Spectrum Systems**, MacMillan, 1985.

15. Viterbi, A. J. and Omura, J. K., **Principles of Digital Communication and Coding**, McGraw-Hill Book Company, 1979.

16. Odenwalder, J. P., "Optimal Coding of Convolutional Codes," Ph.D. dissertation, University of California, Los Angeles, 1970.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Department Chairman, Code 62 Naval Postgraduate School Department of Electrical and Computer Engineering Monterey, CA 93943-5100	1
4.	Professor P. H. Moose, Code 62Me Naval Postgraduate School Department of Electrical and Computer Engineering Monterey, CA 93943-5100	1
5.	Commander, Naval Ocean Systems Center Attn: Mr. Darrell Marsh (Code 624) San Diego, CA 92152	2
6.	Lt. Kevin S. Hopkins Commander, Cruiser-Destroyer Group THREE FPO San Francisco, CA 96601-4702	3